Journal of Information Systems Education

Volume 32
Issue 2
Spring 2021

Teaching Tip

Design Thinking and Mobile App Development: A Teaching Protocol

Nasser Shahrasbi, Leigh Jin, and Wei-Jun Zheng

Recommended Citation: Shahrasbi, N., Jin, L., & Zheng, W.-J. (2021). Teaching Tip: Design Thinking and Mobile App Development: A Teaching Protocol. *Journal of Information Systems Education*, 32(2), 92-105.

Article Link: https://jise.org/Volume32/n2/JISE2021v32n2pp92-105.html

Initial Submission:	5 August 2020
Accepted:	13 January 2021
Abstract Posted Online:	13 March 2021
Published:	9 July 2021

ISSN: 2574-3872 (Online) 1055-3096 (Print)

Teaching Tip
Design Thinking and Mobile App Development: A Teaching Protocol

Nasser Shahrasbi
Leigh Jin
Information Systems Department
San Francisco State University
San Francisco, CA 94132, USA
shahrasbi@sfsu.edu, jinlei@sfsu.edu

Wei-Jun Zheng
University of Wisconsin at Parkside
Kenosha, WI 53144, USA
zheng@uwp.edu

ABSTRACT

The growth in the tech industry in recent years has increased business major students' interest in software programming and app development. However, the traditional way of teaching these courses involves intensive coding exercises and little interaction among students. These methods often discourage the students due to the slow learning curve and limit instructors' options to engage the students in class. To address these problems, we designed a teaching protocol for integrating Design Thinking into a Mobile App Development course. Our protocol offers a step-by-step implementation guide and instances from a real-life examination at a California State University. The proposed guideline can be extended and applied to other courses and curriculum in the business information systems domain.

Keywords: Design thinking, Mobile application development, Curriculum design & development, Project-based learning, Experiential learning & education, Teaching tip

1. INTRODUCTION

The tremendous growth in the mobile app industry and app usage in recent years has made app development one of the most in-demand skills in the U.S. (Gallaugher et al., 2017). A recent survey answered by more than 2,000 CIOs from around the world reports high demand for app development skills in both enterprise and consumer markets (Columbus, 2016). About 80% of the respondents claim high demand for app developers as app usage grows in their companies and among their customers. These findings are in line with the rapid expansion of mobile apps in the consumer market. Companies like Apple, Google, and Amazon are reporting record numbers of app downloads. For example, the number of apps available through the Apple App Store has grown exponentially in less than 12 years, from only 500 apps in 2008 to more than 2.2 million apps in 2020. The "iOS app economy" has created 4.5 million jobs globally and brought in $189 billion to iOS app developers (Statista, 2020).

In this context, Design Thinking (DT) has been recognized as a methodology that promotes the design of engaging and user-friendly apps. DT is a relatively novel design perspective that offers important recommendations for innovation and creativity (UK Design Council, 2010). It follows a human-centered approach that involves different stakeholders, including end-users and customers, in the design process. Companies such as Apple, IDEO, Microsoft, and SAP follow DT guidelines and principles to increase user empathy, passion, engagement, and creativity in their products, services, and app development processes.

Along with its growing impact on businesses, DT has attracted special attention in education. Several attempts have been made to introduce DT to academia and especially to teaching and learning (Beckman and Barry, 2007; Palacin-Silva et al., 2017). While some initiatives such as the Stanford d.school (Institute of Design at Stanford University) and the Hasso-Plattner-Institute d.school in Potsdam Germany have designed special programs for educating both students and educators with the basic concepts and principles of DT, others

have tried to expand the applications of DT in different curricula (Beckman and Barry, 2007; Wong, 2009; Zupan and Nabergoj, 2012; Kelly and Kelley, 2013). While these efforts have made tremendous contributions to advancing DT applications in many fields, such as engineering and design, this perspective has yet to grow in business, and specifically information systems, curriculum. One of the main reasons for this seems to be the lack of a well-defined template or guideline for faculties and instructors on how to integrate DT into their courses and curriculum (Wong, 2009; Palacin-Silva et al., 2017). To fill this gap, we designed a guideline (teaching protocol) that offers a step-by-step guide for educators and instructors to successfully implement and transform traditional mobile app development classes with DT.

The rest of the paper is organized as follows. We first explain the principles of DT and elaborate on how these principles can be implemented in the classroom context. We then explain our guideline, showcasing a redesign of a mobile application development course at a California State University institution, and discuss the effectiveness of the implemented changes.

2. BACKGROUND

2.1 The Concept

The concept of DT has evolved over the past couple of decades (Micheli et al., 2019). It is broadly applied to a school of thought which trusts in its capacity to foster creativity and innovation in organizations (Brown, 2008). According to the UK Design Council (2010), DT is a method to understand problems and offer innovative solutions. It is a systematic perspective (process) to help understand the nature of the problem and provide solutions that are more compatible with the user's desires, needs, and wants. Johansson-Sköldberg, Woodilla, and Çetinkaya (2013) distinguish between the term "designerly thinking" and what is known as "DT," describing the former as the designer's professional, non-verbal, and practical skills and competence to create new things and the latter as the process by which design takes place. DT involves the engaged and active participation of all the various stakeholders in the design process as they discuss and share ideas about the problem and solution. DT also involves several iterations in which the design team expands or limits the options, ideas, and solutions, which are also known as "diverge" and "converge" (Lewrick, Link, and Leifer, 2018). DT is based on prototyping and rapid testing of the ideas and solutions that allow the design team to test different ideas and embrace good solutions while ruling out the nonviable ones.

2.2 Principles and Process

According to the Stanford d.school, DT has five main steps or phases: empathize, define, ideate, prototype, and test (see Figure 1).

DT is based on truly understanding and realizing the problem. Therefore, the first step of the DT model is to Empathize, gaining an understanding of user pain points and problems. Design thinkers often need to engage multiple research methods, including interviews, observations, and ethnographic immersion, to gain a deeper, personal understanding and insight into users and their needs. During the Define phase, the information gathered in the previous stage is analyzed and synthesized until the team reaches an agreement and a common understanding of the core problems that need to be solved. The Ideate phase requires the team to brainstorm and try to find as many ideas as possible regardless of how practical or realistic the ideas are. However, eventually, the team needs to prioritize and settle on the set of best ideas that they would like to carry into the Prototype phase, during which the DT team will try to quickly produce several physically tangible, inexpensive, simulated versions of the products to experiment and test with users. Finally, in the Test phase, the best solution features that are identified during the previous stages are combined and rigorously tested.

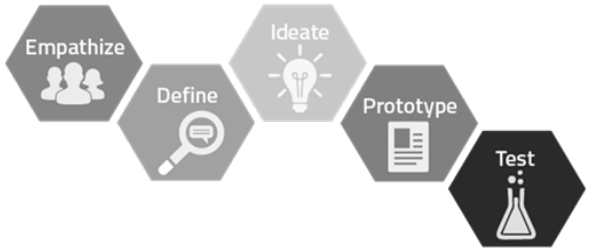


Figure 1. Stanford d.School's Process of Design Thinking

As a new way of thinking, DT offers guidelines and principles that are different from other design methodologies and techniques. Here are some examples:

1. Belief in creativity not as a talent, but rather a process and a skill that can be acquired through practice.
2. Reliance on interdisciplinary teamwork.
3. Focus on design for user experiences instead of just design for products or systems.
4. Encouragement to fail quickly and cheaply to succeed.
5. Emphasis on early experimentation through low-fidelity prototypes and visualization techniques.
6. Involvement of users early and often, incorporating their needs and feedback through iterative prototyping and testing.

2.3 Impact on Education

Integrating DT in education (especially in teaching and learning) can provide several benefits for both students and educators. Mentzer, Becker, and Sutton (2015) argue that DT can reinforce multiple skills in students, including critical thinking, problem-solving, and creativity. It also helps students develop social skills, such as communication, empathizing, and collaboration. Zupan and Nabergoj (2012) argue that entrepreneurship education has been demonstrated to be ineffective, which has motivated practitioners to explore opportunities to use DT for problem-solving. One of the driving motivations for adopting DT is the potential gain of "creative confidence" capability (Lee and Raghu, 2014). Lee and Raghu (2014) suggest that the level of creative confidence can be improved through practice when skills like empathy and capability are available and well-practiced. Appropriately packaging and teaching these skills to students contribute to their creativity, which can lead to developing creative

confidence. Glen, Suciu, and Baughn (2014) emphasize the need to incorporate DT into business courses, arguing that DT has traditionally been adopted and taught in design-related classes and fields, such as product design, engineering, architecture, and urban design. However, adding and integrating DT principles and processes can strongly impact the quality of the teaching and learning experience in business courses.

Apart from its implications for students, DT has tremendous benefits to instructors. Scheer, Noweski, and Meinel (2012) argue that DT in class can improve the learning environment for both instructors and students while facilitating constructive learning and addressing students' individual and collective work. Retna (2016) reported that teachers' perceptions and experience from adopting and integrating DT in their courses were highly positive and encouraging despite the challenging implementation details. They also suggest that DT improves students' engagement and class participation and reduces withdrawal rates and class absences.

2.4 The Gap

Despite these implications, DT still lacks enough penetration in education and especially curriculum design (Wong, 2009). One reason for this that is recognized by many is the lack of a unified teaching protocol to guide educators on how to integrate DT into their curriculum (Beckman and Barry 2007; Wong, 2009; Valentim, Silva, and Conte, 2017). Valentim, Silva, and Conte (2017) argue that many DT courses merely focus on teaching the concepts and underlying principles rather than incorporating those principles into their curriculum. This is especially important for DT courses as the learning occurs through experiencing, rather than merely learning, concepts. Wong (2009) suggests the gap between DT application in education and industry stems from the lack of a unified teaching model in different fields: "[Thus,] no one can agree on how to teach its methods."

3. MOBILE APP DEVELOPMENT: A TEACHING PROTOCOL WITH DT

This section discusses our teaching protocol or guideline. We explain our teaching protocol by showcasing a course redesign of a mobile app development course using DT principles and guidelines. Our goal here is to discuss the underlying pedagogical approaches, teaching tools, class schedule, and implementation process through illustrating a concrete example of how DT principles and guidelines can be assembled to enhance course development in the IT business curriculum.

3.1 Pedagogical Approaches

This course is designed based on the premises of two well-established pedagogical approaches:

3.1.1 Project-based learning. Project-based learning (PBL) refers to a mode of education in which students apply conceptual knowledge to real-life contexts in the form of a project (Markham, 2011). In PBL, students embark on a real-life challenge/problem and find creative solutions by working individually or in teams. This helps students develop not only a deeper understanding of the studied concepts, but also skills, such as critical thinking, teamwork, interpersonal/social skills, communication, leadership, and so forth. The PBL process entails lots of discoveries, experimentation, and hands-on activities that encourage students' innovation and creativity. In this method, the role of a teacher is more as a coach or guide.

3.1.2 Experiential learning. Experiential learning (ExL) or "learning by experience" is built on the premise of learning-by-action (Beard, 2010). ExL includes a series of teaching techniques that motivate students to explore the application of learned concepts in real-life contexts or a simulated environment (Beard, 2010; Felicia, 2011). The method applies techniques that engage students in hands-on activities and exercises to reinforce a positive learning experience.

Both ExL and PBL are popular approaches in contexts where the goal of teaching and learning is to develop skills or learn the application of complex concepts in the real world. The two methods promote active learning and interaction among students, giving them an active role in their learning process (Greeno, 2006). In this course, we use ExL and PBL to a great extent in that all students must choose a real-life challenge to work on in teams during the course. Each class includes activities and hands-on exercises that allow students to apply the learned concepts and techniques in the course to the context of their projects. The designed activities follow a logical path to an outcome solution for students' chosen problem.

3.2 Pedagogical Tools

There are multiple software and pedagogical tools that can help facilitate the implementation of the activities in this course. Below, we describe some of the tools that we used in our course. However, alternative tools can be used by instructors depending on their needs and on availability.

- **Moodle:** We used Moodle (a learning management system) to share the class content and material, including the textbook, PowerPoint presentations, recorded videos and lectures, exercises, assignments, and other resources, with the students. Students are expected to read the material and learn about the concepts before class. This saved a considerable amount of class time for class activities and hands-on exercises. We also used online forums and discussion rooms to provide a platform for the students to create teams, communicate, and brainstorm ideas virtually.
- **Mural:** Mural is a digital workspace that facilitates students' collaboration in their DT activities. Mural provides a digital platform for students' collaboration in and outside of the classroom. It allows students to capture their work and digitally portray their ideas inside and outside of the class as well as digitally capture their design ideas and visually portray the status of their work at the end of each class so they can continue to work on it outside the class or pick up where they left off the next time they meet. In addition, it makes the presentation and sharing of the results much easier and more aesthetically pleasing.
- **InvisionApp:** InvisionApp is a wireframe prototyping tool for mobile application development. The software

allows the student teams to quickly prototype the look and feel of mobile app screens. It can transform static wireframe designs into interactive prototypes through hotspot linking. In addition, students can easily share and test their prototypes with users through a URL.

- **Xcode:** Xcode is Apple's Integrated Development Environment (IDE). It allows students to implement iOS apps in Swift language and helps manage the editing, testing, and debugging of codes within a single workspace. The Xcode interface builder (UI Kit)/live preview (SwiftUI) features a visual design editor that graphically connects the objects and navigation components. It is a smart IDE designed with app developer experience in mind. While teaching iOS development occupies some class time, this skill helps students to implement their design prototype and deliver a minimum viable product of their mobile app. In addition, iOS development skill is highly valued in today's job market. However, if iOS is not the development choice of the instructor, a different platform could be taught to implement mobile applications.

3.3 Team Building

This course is designed with an emphasis on teamwork and collaboration. While team building is explained as an important stage of the process below, the structure and dynamics of the teams are important aspects of running a successful DT course. Project teams of four or five students are formed early in the semester based on their background, skills, and interests in project contexts. In this course, we used a tool called "CATME" to form and monitor teams' dynamics and to calculate individuals' contributions in teams. CATME is a free online peer evaluation tool for a comprehensive assessment of team member effectiveness (http://www.catme.org). The tool is designed based on psychology and education theories, which allows instructors to create and administer "customized, research-based peer evaluation surveys" in their courses. Assigning team members in consideration of their different roles and interests is an imperative factor. Each team must appoint clear roles for the members and assign responsibilities. In general, we considered four roles/skills in teams to be imperative: project manager or coordinator, documenter, designer, and app developer. While these roles do not necessarily define the responsibilities and workload of each member, they can help create a good combination and diversity of roles and interests in teams. Their responsibilities are summarized as follows:

- **Project Manager:** The project manager is responsible for managing the schedule and scope of the project. They determine the timing of each activity, delegate tasks, motivate the team, and resolve conflicts to ensure the progress of the team. This role is especially important in the classroom setting where no dedicated DT facilitator or coach is assigned to teams. This person should also set up accounts in Slack, Mural, and InvisonApp for the entire team.
- **Documenter:** The documenter is responsible for capturing the team's work (e.g., Post-its on the wall) and digitizing it so that the content can be easily shared. This person is also responsible for taking notes, snapping photos, writing up summaries, and documenting the progress of the team. While this role may not seem glamorous, it is essential to ensure a successful DT project. Over the years, we have seen a lot of creative ideas that were left on the wall and never used in later prototypes, presentations, or reports.
- **Designer:** This role is in charge of UI design. The designer helps the team visualize ideas and manipulate digital assets or graphics. Understanding how to use tools like Photoshop or Sketch is a plus; these skills are particularly handy for wireframe design during high-fidelity prototyping of mobile apps.
- **App Developer:** This role is responsible for implementing a subset of the prototype features in an executable iOS mobile application which can then be deployed to run on an iOS device. Some prior programming language experience is helpful in this role. Of course, this role may need to implement the minimum viable product in a programming language other than Swift if the instructor decides to teach a different app development platform in the class.

We encouraged students to take turns or switch roles throughout the course so that they could explore different interests and develop skills and empathy in different roles.

3.4 Class Schedule and Implementation Process

This course is designed for a 14-week, 3-hour class schedule (see Appendix A). In general, each class starts with the instructor's introduction or review of the class material. This usually takes no more than 15-20 minutes and continues with the students' summary of their progress (15-20 min). The students are then assigned a hands-on activity or exercise relevant to the topic of the class, which most often requires them to apply a learned concept to an aspect of their project (45-60 min). If the student teams cannot finish the hands-on activities in class, they will be encouraged to meet outside of class to complete the rest and submit the deliverables on Moodle. For the rest of the class, the instructor teaches iOS app development on Xcode (75-90 min). In this paper, we will focus our main discussion on how to adapt and integrate the DT curriculum into a traditional programming course. Therefore, the app development curriculum is not our main concern here. For iOS development, we follow the textbook *App Development with Swift* (Apple Education, iBook). Some main topics covered include language syntax, UI components, navigation, tables, maps, and JSON parsing. Instructors can choose different textbooks or app development platforms to introduce the core concepts.

The implementation process includes two phases: preparation and execution. During the preparation phase, preparing the class setting, project definition, and team building take place. The execution phase includes benchmarking, user research, synthesis, prototyping and testing, and finally implementation and deployment.

3.5 Preparation Phase

3.5.1 Space and material preparation. Preparing classroom settings, space, and materials is one of the most important

things and should be considered early in the course. In general, the traditional classroom does not provide an ideal setting for DT exercises and activities. This is mostly because both the instructor and students require workspaces for brainstorming and activities, game tables, writing surfaces, and so forth. In an ideal setting, DT space is often a flexible and easy-to-configure setup with writable walls, whiteboards, markers, sticky notes, and other required tools to successfully run the required activities. However, most traditional classrooms/labs in public universities are designed to hold as many students as possible. Tables or chairs are often anchored to the floor, and there is not much free space left to support student teamwork. Figure 2 compares the space arrangements in a DT class (top picture) setting versus a traditional computer lab (bottom picture).

Stanford d.school

Traditional Computer Lab

Figure 2. Classroom Setup for DT Activities

Witthoft and Doorley (2012) in their book *Make Space: How to Set the Stage for Creative Collaboration* describe the required materials and spaces to run a DT and innovative classroom and suggest several creative ways to compensate for the lack of materials or ideal setting. For instance, if you lack enough space in the class, you can use the hallway or the common areas to set up new working spaces for the students (see Figure 3). You can also use self-stick note pads or static cling dry erase sheets to compensate for the lack of writing surfaces in the classroom. In our experience, even though static cling dry erase sheets are slightly more expensive, they are erasable, reusable, and more compact to carry. After each exercise, students can roll up their working sheets to take with them so that the classroom is ready for the next class.

Figure 3. Class Settings Examples

In addition to space arrangements, DT exercises may also require material preparations that are not typically available in a traditional classroom setting. For example, Post-it notes are extensively used for brainstorming and organizing ideas into meaningful categories. Colorful dot stickers are helpful for students to place votes on different ideas. Because timeboxing is essential for almost all DT exercises, a timer or timer app is required to record timed activities. Below is a list of required materials for running DT activities in class:

- Post-it easel pad/static cling dry erase sheets help to quickly turn a basic wall into a writable surface.
- Post-it notes are one of the signature DT supplies that help students share and organize their teamwork ideas through the creative process.
- Markers and Sharpies make writing easy to read and share in class.
- Dot stickers allow team members to vote on favorite ideas and prototypes.
- Letter-size paper can be used for low-fidelity sketches of mobile app screens.
- A timer facilitates visual time management during the DT activities.

3.5.2 Define challenge. The second step involves exploring and selecting real-life challenges. At this stage, students are required to think about and explore real-life challenges to find the right topic or context for their team project. The main goal here is to allow students to explore real-life challenges without thinking about the solutions. Three approaches can be followed to help students identify the challenges they would like to work on:

- **Instructor-led:** In this approach, the instructor provides a list of predefined challenges to the students. Alternatively, the instructor can provide access to certain publicly available datasets (e.g., https://www.data.gov/) and ask the students to select a challenge based on their interest. The United Nations

Global Issues (https://www.un.org/en/sections/issues-depth/global-issues-overview) is another excellent source of real-life challenges. In addition, downloadable datasets and real-time or live-update data APIs can help to inspire creative app ideas. For example, the SF Bart API (http://api.bart.gov/ provides detailed information on train stations, routes, and real-time schedule estimates, offering a great opportunity for students to ideate creative apps.

- **Student-picked:** This approach lets students freely explore and identify their challenges to work on. It may not be ideal for large classes where the instructor may not have sufficient time to guide individual students. Therefore, what is often suggested is to define a general context or category of topics in which the students can suggest comparable projects with the same level of difficulty and complexity.
- **Company-led:** The instructor could also contact companies or non-profit organizations for industry-specific challenges. This is a win-win situation because the DT project is a great opportunity for these organizations to recruit talent among students.

Some important factors to consider in project selection include creativity, value, and ethics. A challenge should be creative in the sense that it should not be a problem that has been solved already. It should also deliver value to investors. In other words, its target audience or impacts should be broad enough to make the app development a worthwhile effort. Finally, it should be ethical. For example, a challenge that promotes narcotics consumption should not be encouraged in class.

3.5.3 Team building. It is crucial to form diverse and interdisciplinary teams for effective DT practices. Assuming students have come up with a number of interesting design challenges, the instructor can give them some time to discuss those challenges they are most interested in. Students are asked to pick roles (e.g., project manager, documenter, designer, app developer, etc.) that best describe themselves by selecting the corresponding colored dot stickers. After writing down their names on the dot stickers, students can then place the stickers next to the challenges they are most interested in joining. The challenges will be ranked based on the number of dot stickers received.

Another round of voting could be conducted after eliminating challenges with the least votes. The students will adjust their choices of challenges and roles accordingly until teams are formed around a set of challenges. This exercise helps students to visualize their team structure and to negotiate their roles based on the team's needs. After each team is built, it is beneficial to conduct team-building exercises to help students become familiar with their teammates. In addition, teams are required to create slack channels to facilitate team communications.

3.5.4 Reframe challenge. After teams are formed and general challenge topics are chosen, it is time for some scoping. During this stage, each team is required to discuss and reach a common understanding of the challenge they are willing to solve. The students are given opportunities to modify or rephrase the challenge they selected earlier. For example, they could identify the general problem and target audience by filling in the blanks in a sentence like the following:

> *We would like to develop a solution for _____(a social / environmental / enterprise / civic challenge), more specifically, we would like to improve the experience for people who _____.*

Below is an example of a reframed challenge:

> *We would like to develop a solution for enhancing walking safety, more specifically, we would like to improve the experience for women travelers.*

The deliverable for each team would be a list of team members (and their roles) along with a broadly framed problem statement. Therefore, by the end of this preparing phase, the instructor should be able to assemble student teams around a set of defined DT projects that they have agreed to work on.

3.6 Execution Phase

3.6.1 Benchmarking/market research. The market research stage helps students gain a deeper understanding of the problem by researching, exploring, and collecting more information. The goal is to make sense of market trends and existing or similar solutions through additional analysis of data, products, or services. The outcome of this research will help teams to not only identify potential opportunities for innovation but also to avoid prototyping duplicate solutions.

For these purposes, the Google search engine is a great place to start. Two main activities include bibliographic/desktop searches and app store searches. Desktop search refers to running a web search about the investigated phenomenon and collecting information about the unseen facets of the problem. Students can run Google searches or use other search engines to learn about the concepts or questions around the investigating phenomenon. In doing so, it is helpful to ask the students to search for academic research in databases or even Google Scholar to find rigorous academic research on their topic of interest. The instructor can encourage students to also explore other resources, such as tech blogs, YouTube, or social media websites, to get inspired not only about the solutions but also about the context and problems they are investigating. The result of the desktop search should be summarized, shared, and presented in class, and the instructor could also encourage students to present their findings using images and infographics for better visual communication. The following is an example of students' research on parking problems in San Francisco (Figure 4).

Figure 4. Students' Project Benchmarking Sample

App store research involves exploring the iOS or Android app stores for existing mobile apps and their features that are attacking similar problems/issues. Students should identify some of the problems with existing apps and explain why they need more innovation in this problem space. Of course, the results of this step should also be summarized and presented in class (formally or informally). Below is an example of a student's App Store search project result on existing apps that potentially help to solve the parking problem (Figure 5).

Figure 5. Students' Project Solution Research Sample

3.6.2 User research. User research includes activities that help students understand the users' needs, behaviors, attitudes, and experiences. User research can be done through various qualitative and quantitative methods, including observation, interviews, focus groups, experiments, surveys, etc. In a classroom setting, user interviews and observations are the most important DT activities to help students empathize with users. The goal is to gain a first-hand understanding of the true needs of their target audience. There are multiple resources for learning how to conduct user interviews and observations. Indeed, these are well-defined methods in both qualitative research and User Experience (UX) research protocols. Specifically, Userpalooza by Bowmast (2018) is very practical in providing detailed guidelines on how to plan and conduct field research to inform design projects.

However, these methods are not typical classroom learning experiences for most students. Therefore, some in-class preparation and practice before sending teams out to the field would be beneficial. Before conducting interviews, students should draft an interview guide that includes 5-8 general interview questions. These questions should focus on exploring how and why users choose to solve their problems in certain ways and their main pain points in the process. Table 1 provides a sample interview guide regarding a challenge related to insurance. In addition to interview questions, students should also determine where to find their target users and how they will form interview teams. In general, interview teams should consist of at least two members. While one interviewer leads and conducts interviews, the other member(s) should observe, take notes, and support the interview process. If time permits, the instructor could also encourage different student teams to interview each other to practice their interview skills.

Questions	Answers
User demographic profile	
Name: Gender: Age: Education: Occupation:	
User experience, needs, pain points, and current tools	
Do you have insurance? What kind?	
What comes to your mind when you think about your insurance?	
Why is that?	
How would you describe the "ideal insurance"?	
Please describe an unpleasant experience dealing with insurance companies?	
Why are you happy/unhappy with your current insurance provider?	
What would you like your insurance provider to do differently? Why?	
What tools are you using currently to manage your insurance?	
How does it work or does not for you?	

Table 1. User Interview Form Sample

After paired interview teams are formed, students can visit places on campus (such as the cafeteria, student dorms, and bookstores) or surrounding areas (such as malls or public transit stations) to conduct interviews. On average, each user interview should take about 10 minutes, and each interview team is expected to interview 5 users in about 1 hour. In addition to an interview activity during class time, student teams are also encouraged to interview more users after class. In the end, each team should summarize and share their user research findings. Table 2 provides an example of a user research summary.

Profile	Occupation, Insurance type	Key Quotes from Interview
Alice/F/25	-working on-campus -car insurance -international student health insurance	- *"I would prefer if someone just took a statement from me rather having to take multiple calls to explain the situation"* - *"Insurers are stressful to deal with in America when English is not your first language"* - *"I felt even harassed by the insurance agent, when trying to explain the car incident"*
Ken/M/27	-auto shop -business insurance -personal health -car insurance	- *"Fairness and an easy to understand policy is what I look for when choosing an insurer"* - *"I prefer an all in one solution"* - *"I am willing to pay more if an insurer is willing and able to solve problem quickly and easily"*
Greg/M/28	-software engineer -renters insurance	- *"Maybe you guys could insure office furniture?"* - *"Are lost items covered by insurance?"* - *"People would take advantage of the insurance, right?"* - *"ML model to predict how likely someone is to claim insurance?"*

Table 2. User Interview Summary Sample

3.6.3 Synthesis. After extensive user research, it is time to revisit the problem itself and narrow down its scope into more specific domains. During this stage, which is also known as "synthesis," the students should derive insights about the true needs and experiences they would like to improve for their target users. The goal here is to create user "Personas" and problem "Points of View" to inform the direction of the project. They provide the overall vision of the solution design by answering two important questions: who is our target user and what is the exact problem we are solving here? A persona is essentially an aggregated profile of a specific user type. It is important to make this user as vivid as possible, including name, age, education, goal, needs, and pain points (https://www.invisionapp.com/inside-design/user-persona-template/). In most cases, the persona profile could be assembled from a user or multiple users whom students have interviewed earlier.

The persona is important because it helps guide the design decisions in the later stages of DT. To define personas and points of view, students could take turns sharing user stories they discovered during their interviews. The various aspects of user needs are captured on Post-it notes, which are subsequently regrouped into meaningful categories based on similarities or relationships among concepts. Some teams may experience difficulties in reaching an agreement in the synthesis stage, but this struggle is necessary for the team to develop a common understanding of the overall vision of the project. Instructor coaching and encouragement can be extremely helpful at this phase. When students must choose one solution among many alternatives, they often look back into the persona profile and it becomes instantly clear which solution fulfills the needs of their persona most effectively. Figure 6 provides an example of a persona profile.

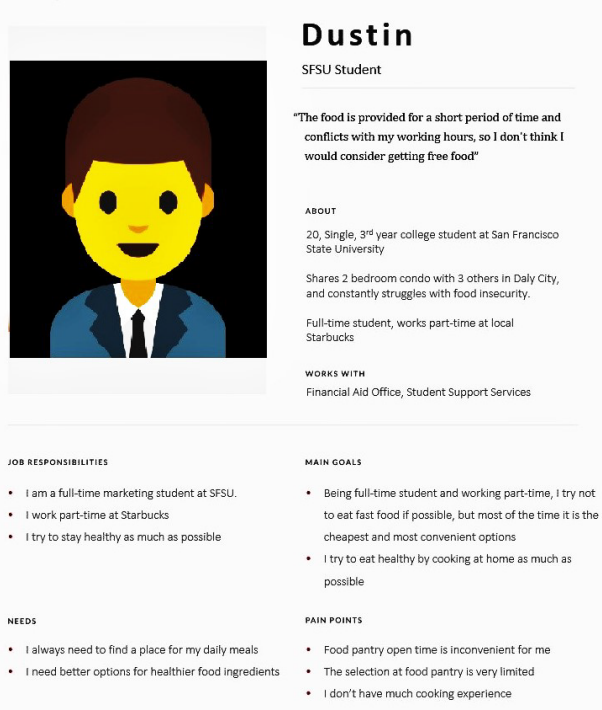


Figure 6. Sample User Persona

Point of view, on the other hand, is essentially a summary statement that captures the persona, the user's needs, and insights generated during the synthesis stage of the DT process. It is a unique way of reframing a design challenge into an actionable problem statement. This statement not only captures the design vision of the entire team but also drives the solution creation process during the ideation and prototyping phases. Point of View can be framed with a simple Mad Lib format according to the Stanford d.school DT Curriculum (2021):

[USER] needs to [USER'S NEED] because [SURPRISING INSIGHT].

Here is an example of a Point of View that students created for a food pantry challenge. It inspired students to design the mobile features to suggest recipes and to allow

students in need to pre-order food pantry items for pickup later at their convenience:

> *A busy working student who struggles with food insecurity needs to access healthy food cheaply and conveniently, but the food pantry service hours do not fit his schedule and he does not know much about cooking*.

3.6.4 Ideation. During the ideation phase, the team officially transits from the problem-defining mode to the problem-solving mode. The key objective here is to generate creative ideas to solve the challenge as defined in the Point of View. To achieve this goal, the Journey Map and Storyboard are two methods with the potential to help students be innovative while keeping focused on their vision. The Journey Map could be considered an extension of synthesis work. It is an excellent way to visualize the experience of the persona through different stages of the journey when they encounter the pursued problem. Most importantly, because it maps out the highs and lows of user emotion shifts from beginning to end, it helps teams to identify opportunities for ideation, especially surrounding the unpleasant moments of the user journey.

Next, the students will brainstorm ideas that can help to improve user experiences through the journey. They can use Post-it notes to generate suggestions for how to relieve user pain points. After these ideas are aggregated and categorized, students can use dot stickers to prioritize and vote on the most important features to be included in their solution. The Journey Map in Figure 7 was created to highlight the student emotions around a current food pantry service and some solutions that might be helpful to improve their experience.

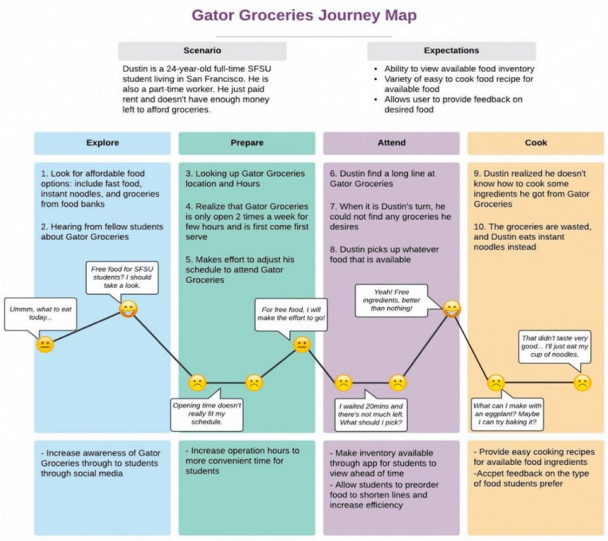


Figure 7. Students' Project Journey Map Sample

After important app features are identified, student teams can create Storyboards to further illustrate when, where, and how these features are used. Storyboarding is a common technique for scene sketching in comic books and the movie industry. In DT, it helps to contextualize ideas and is a powerful way for student teams to show and tell their solutions.

After a student team uses Post-it notes to sketch each frame and assemble their Storyboard, they can present and share it with the whole class. Additional brainstorming sessions can be held to incorporate feedback and improvements. If needed, instructors can ask students to digitize the Storyboard to be included in the final project. Figure 8 is a sample Storyboard from a class project that has been adapted from the Rachel Krause template (https://www.nngroup.com/articles/storyboards-visualize-ideas/).

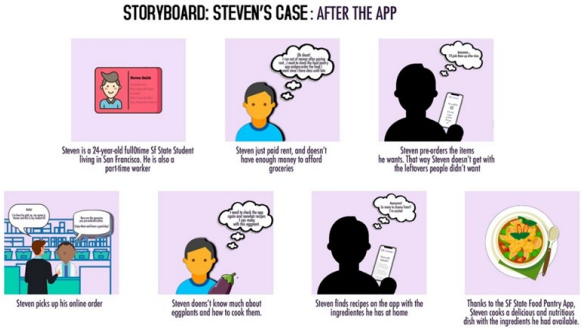


Figure 8. Sample Student Project Storyboard

3.6.5 Prototyping and testing. Prototyping is a core principle of DT. During this phase, students are introduced to a variety of prototyping techniques and methods and asked to develop both low-fidelity and high-fidelity prototypes of their solutions. Prototyping is a common method to get user feedback directly through developing, testing, and improving low-fidelity prototypes of the solutions (Mendoza-Garcia and Cardella, 2014). Prototyping helps to identify and resolve design problems and misunderstandings through iterative and active communication with users.

The prototyping phase includes two steps, which usually take two to three weeks to implement. In the first step, the students will be introduced to the techniques through lectures by the instructor and studying provided materials. Each team then develops and tests low-fidelity prototypes using their selected techniques. For instance, teams can create paper prototypes of the key features of their app, hand-sketching one screen per page. Since this approach helps to create rapid and inexpensive prototypes, students do not hesitate to discard or modify features after user testing. Figure 9 presents a sample low-fidelity prototype from a class project.

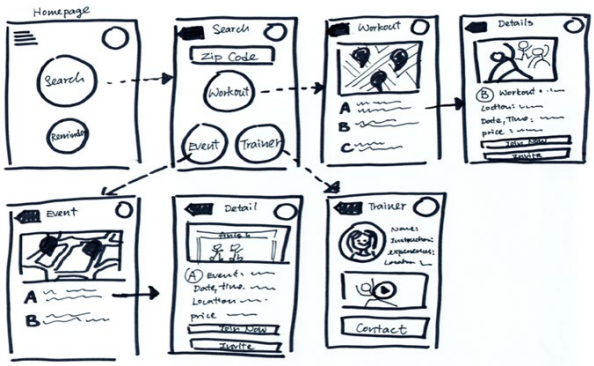


Figure 9. Low-Fidelity Prototype Sample

After a low-fidelity prototype is developed, students can conduct initial user testing with the members of other teams in the class. Mentoring on how to test and improve prototypes can be helpful if offered by the instructor. The prototypes can then be improved based on the given feedback. The next step includes collecting feedback from external users. During this step, the teams can expand their user testing outside the class. Encouraging students to go through multiple rounds of testing is very important because iteration plays a vital role in DT prototyping. According to Mendoza-Garcia and Cardella (2014, p. 1), "teaching students the relevance of [prototyping] iteration will help them to develop skills to become what we can call human-centered informed designers."

After extensive low-fidelity prototyping and testing, students should have captured the key features of their app design. Next, they can proceed with high-fidelity prototyping to include more sophisticated and refined wireframe designs. As mentioned earlier, InvisionApp is an excellent tool for this purpose. Students may also use Photoshop or other software to mock-up their app UI. Figure 10 illustrates a student project sample of a high-fidelity prototype.

Figure 10. High-Fidelity Prototype Sample

3.6.6 Development and implementation. Development and implementation are ongoing processes and take the longest to complete. This stage includes learning mobile app development in iOS Swift language (or other platforms of the instructor's choice). During this stage, students are required to create a functional iOS app to demonstrate the most important features of their team project design in Xcode. They are expected to implement and present at least three to five screens with UI components, including labels, buttons, text fields, images, switches, table views, and navigation controllers. Figure 11 illustrates an example of a functional iOS app that students developed for a food pantry service project. Some students with more technical backgrounds can learn more advanced coding skills through online tutorials and resources provided by the Professor. For example, as shown in Figure 11, the developed app incorporated sophisticated features, including Navigation Controller, Custom Table View, Map Kit, JSON parsing, and Google Firebase, for real-time inventory updates.

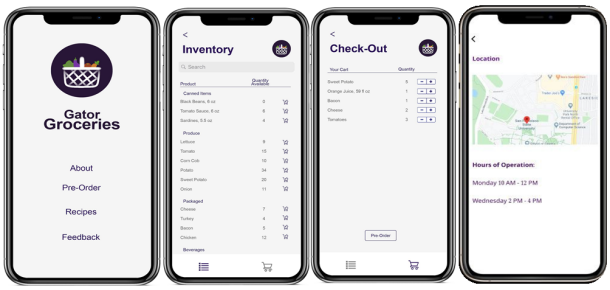


Figure 11. Sample Student Project Final App Interface

After prototyping and implementation, the students will give a final presentation with a working demo of their solutions. The students also need to present their design and development journey and the unique lessons learned from the process. During the presentations, other students can give feedback as the stakeholders and final users of the solution. The students then have one week to improve and submit their solutions for final review.

4. EVIDENCE AND STUDENT FEEDBACK

We applied our teaching protocol in a course redesign of a mobile app development course. After the course, we designed an empirical study and surveyed students who took the class with the new curriculum. A total of 38 students took the course from beginning to end. Based on their survey feedback, we used the Partial Least Squares (PLS) regression method and Smart PLS 2.0 to assess whether the redesigned curriculum successfully achieved the promised outcomes, including students' perceived importance of Design Thinking in mobile app development practices, their learning experience, and their perceived achievement in the course. Our examination of the survey results shows that there are very positive learning outcomes among students. Further analysis shows that the positive learning outcome of our new protocol is from three main psychological drivers: User Empathy, Ownership and Control over Project, and Positive Emotion (Greenson, 1960; Raymond, 2001; Luthiger, 2005).

Although we did not face any major issues during our implementation, multiple challenges may still occur during the implementation phase. One major challenge may be associated with keeping the right balance between the time spent on design thinking and app development/coding activities in class. This is especially important as different student cohorts may have different levels of knowledge and skillsets (technical and non-technical). Therefore, it is important for the instructor to obtain a primary understanding of the students' knowledge of both DT and coding and to reconsider the class material and activity loads accordingly. This can be simply done by a short quiz or assignment at the beginning of the semester. There are many good sources for such quizzes and sample assignments (see Appendix B).

Second, as mentioned above, the Xcode IDE can only be installed on Mac computers and requires the iOS environment. This might be challenging if students do not have Mac computers or laptops. One solution we found effective in our course was to engage with our school's IT department to install more Mac stations in our library and computer labs to give more access and options to the students in our class for

doing their assignments or group activities. In 2020, our library agreed to install Xcode on all Mac laptops that students could borrow from the library and give priority to the students of our class for Mac laptops. In addition, students who were enrolled in iOS app development courses were given priority in checking out these Mac laptops.

Third, as more and more hardware capabilities are packed into iOS devices, both Swift language and Xcode IDE are updating more often and at a faster pace. Sometimes, students' laptops and Xcode IDE automatic updates create confusion and discrepancies in the version and features they use compared to those being used in class or in the computer lab. Therefore, it is important for instructors to take note of the latest updates and changes. In addition, like other IS curriculum, this class demands that instructors constantly update their app development skills to keep up with advances in mobile technology.

5. CONCLUSIONS

DT is not only the missing link between the design and development of software solutions but can also serve as an innovative pedagogical approach that creates incredible opportunities to enhance students' learning and skill development. The literature identifies three groups of benefits and outcomes that can derive from integrating DT into class settings. The first is related to students' learning experience. A traditional software development class is coding-intensive, and students are expected to follow lectures and complete sets of exercises that are specified by their instructors. By introducing DT methods and allowing students to initiate the challenges that they are interested in solving in a team environment, we expect an improved learning experience for students with the new curriculum design. The second outcome is related to the characteristics of the design and development solutions themselves. Besides being creative and innovative, a successful software solution must be desirable, viable, and feasible at the same time (Brown, 2008; Meinel, Leifer, and Plattner, 2011; Johansson-Sköldberg, Woodilla, and Çetinkaya, 2013). Technological feasibility means that there exists a practical technology that makes it possible for the proposed solution to be built soon. Commercial viability assumes that the proposed solution carries the potential to be developed into a profitable business. Last and most important is the social desirability of the solution, that is, the solution must be appealing and enticing to its target users. In other words, no problem has been solved if the solution is not adopted by users. Therefore, we expect the new curriculum design to help students achieve solutions that are more desirable, feasible, and viable to their users. The third outcome concerns the general assessment of the perceived importance of DT as a problem-solving skillset. Through our new curriculum design, we expect to develop DT as an important skill for students' careers, personal life, and studies. The perceived importance in all three areas may imply that students would transfer DT knowledge across different domains of their lives.

We hope that this study highlights these opportunities and encourages more educators in the community to integrate DT into their curriculum. Future researchers can discover further opportunities and empirically test the impact of DT on information systems education.

6. REFERENCES

Beard, C. (2010). *The Experiential Learning Toolkit: Blending Practice with Concepts*. Kogan Page Publishers.

Beckman, S. & Barry, M. (2007). Innovation as a Learning Process: Embedding DT. *California Management Review*, 50, 25–56.

Brown, T. (2008). Design Thinking. *Harvard Business Review*, 86(6), 84-92.

Bowmast, N. (2018). *USERPALOOZA - A Field Researcher's Guide*. Nick Bowmast Publishers.

Chin, W. W. & Nested, P. R. (1999). Structural Equation Modeling Analysis with Small Samples Using Partial Least Squares. In R. H. Hoyle (ed.), *Statistical Strategies for Small Sample Research*. Thousand Oaks, California: Sage.

Columbus, L. (2016). 63% of Companies Operating in the Cloud can Develop an App in 3 Months or Less. *Forbes Tech Report*. Retrieved on February 25, 2016, from http://www.forbes.com/sites/louiscolumbus/2016/02/24/63-of-companies-operating-in-the-cloud-can-develop-an-app-in-3-months-or-less/#282aea8313ff.

Doorley, S. & Witthoft, S. (2012). *Make Space: How to Set the Stage for Creative Collaboration.* John Wiley & Sons.

Efron, B. & Tibshirani, R. (1993). *An Introduction to the Bootstrap.* New York, New York: Chapman Hall.

Gallaugher, J., Cho, S., FitzGibbon, J., & Mahecha, J. A. (2017). The Flipped Classroom and Mobile App Development: A New Model for Engaging Students in a First Programming Course. In *Proceedings of the Twenty-third Americas Conference on Information Systems*, Boston, Massachusetts.

Glen, R., Suciu, C., & Baughn, C. (2014). The Need for Design Thinking in Business Schools. *Academy of Management Learning and Education*, 13(4), 653-667.

Greeno, J. G. (2006). Learning in Activity. In R. K. Sawyer (ed.), *The Cambridge Handbook of the Learning Sciences*, New York, New York: Cambridge University Press, 79-96.

Greenson, R. R. (1960). Empathy and its Vicissitudes. *The International Journal of Psychoanalysis*, 41, 418-424.

Felicia, P. (2011). *Handbook of Research on Improving Learning and Motivation through Educational Games: Multidisciplinary Approaches*. Hershey, Pennsylvania: IGI Publishing.

Johansson-Sköldberg, U., Woodilla, J., & Çetinkaya, M. (2013). Design Thinking: Past, Present and Possible Futures. *Creativity and Innovation Management*, 22(2), 121-146.

Kelly, T. & Kelley, D. (2013). *Creative Confidence: Unleashing the Creative Potential within Us All*. Crown Business.

Lee, G. & Raghu, T. S. (2014). Determinants of Mobile Apps' Success: Evidence from the App Store Market. *Journal of Management Information Systems*, 31(2), 133-169.

Luthiger, B. (2005). Fun and Software Development. In *Proceedings of the First International Conference on Open Source Systems*, Genova, 11-15.

Lewrick, M., Link, P., & Leifer, L. (2018). *The Design Thinking Playbook: Mindful Digital Transformation of Teams, Products, Services, Businesses and Ecosystems*. John Wiley & Sons.

Markham, T. (2011). Project Based Learning. *Teacher Librarian*, 39(2), 38-42.

Meinel, C., Leifer, L., & Plattner, H. (2011). *Design Thinking: Understand-Improve-Apply*. Springer.

Mendoza-Garcia, M. E. & Cardella, M. E. (2014). Using Alien Centered Design for Teaching Iteration in The Design Process in Undergraduate Design Courses. In *Proceedings of the IEEE Frontiers in Education Conference (FIE)*, 1-8.

Mentzer N., Becker, K., & Sutton, M. (2015). Engineering Design Thinking: High School Students Performance and Knowledge *The Research Journal for Engineering Education*, 104(4), 417-432.

Micheli, P., Wilner, S. J., Bhatti, S. H., Mura, M., & Beverland, M. B. (2019). Doing Design Thinking: Conceptual Review, Synthesis, and Research Agenda. *Journal of Product Innovation Management*, 36(2), 124-148.

Palacin-Silva, M., Khakurel, J., Happonen, A., Hynninen, T., & Porras, J. (2017). Infusing Design Thinking into a Software Engineering Capstone Course. In *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*, 212-221.

Raymond, E. (2001). *The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly Media.

Retna, K. S. (2016). Thinking About 'Design Thinking': A Study of Teacher Experiences. *Asia Pacific Journal of Education*, 36(sup1), 5-19.

Scheer, A., Noweski, C., & Meinel, C. (2012). Transforming Constructivist Learning into Action: Design Thinking in Education. *Design and Technology Education: An International Journal*, 17(3), 8-19.

Stanford d.school DT Curriculum. (2021). Tools for Taking Action. Retrieved April 1, 2021, from https://dschool.stanford.edu/resources.

Statista. (2020). Number of Apps Available in Leading App Stores as of 2nd Quarter 2020. *Statista - The Statistics Portal*. Retrieved January 1, 2021, from https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/.

UK Design Council. (2010). Multi-Disciplinary Design Education in the UK. Retrieved April 1, 2021, from http://www.designcouncil.org.uk/sites/default/files/asset/document/multi-disciplinary-design-education.pdf.

Valentim, N. M. C., Silva, W., & Conte, T. (2017). The Students' Perspectives on Applying Design Thinking for The Design of Mobile Applications. In *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET)*, 77-86.

Wong, V. (2009). How to Nurture Future Leaders: World's Best Design Schools. *Bloomberg Businessweek*. Retrieved January 1, 2021, from http://www.businessweek.com/innovate/di_special/20090930design_thinking.htm.

Zupan, B. & Nabergoj, A. S. (2012). Developing Design Thinking Skills in Entrepreneurship Education. *International Design Management Research Conference*, Boston, Massachusetts, 525-535.

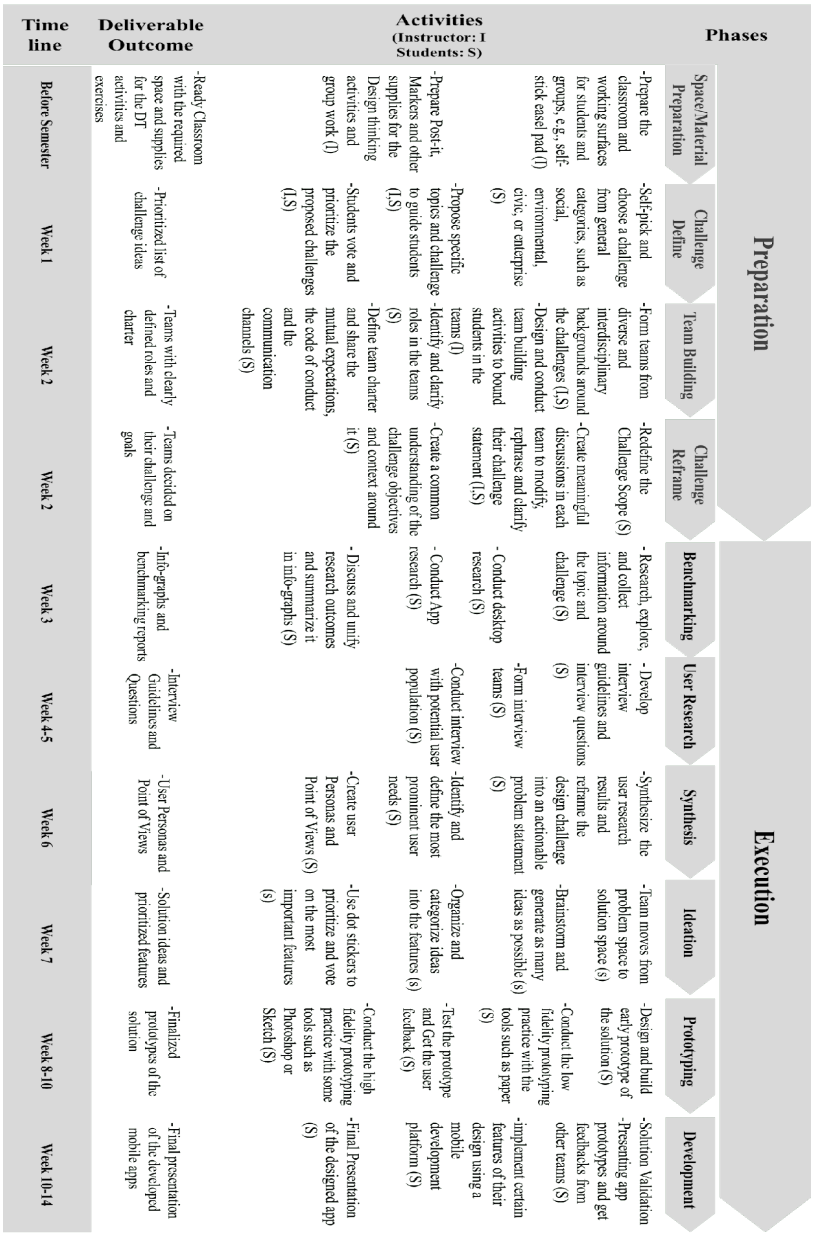
AUTHOR BIOGRAPHIES

Nasser Shahrasbi is an assistant professor of information systems at San Francisco State University. He received his Ph.D. in information systems from HEC Montreal. His research insterest include digital transformations, information systems education, design thinking, and experiential learning. He has published several articles in journals and conference proceedings, including the *Journal of Supply Chain and Operations Management*, *Journal of Marketing and Strategic Management*, and the *International Journal of Entrepreneurship and Small Business*.

Leigh Jin is a professor of information systems at San Francisco State University. She earned her doctorate in computer information systems from Georgia State University. Her research interests include technology innovation with design thinking, enterprise mobility, mobile reputation systems, open source software adoption, and virtual organization/community.

Wei-Jun Zheng is an associate professor of information systems at University of Wisconsin Parkside. He received his Ph.D. from University of Oklahoma. His research interest include IS auditing and big data analytics, IS outsourcing, mobile reputation systems design, and IS strategy. He has published in *MIS Quarterly*, *MIS Quarterly Executive*, *Journal of the Association for Information Systems*, *IEEE Transaction on Engineering Management*, *Information Resources Management Journal*, *Information Systems and e-Business Management (ISeB)*. and many national and regional conference proceedings.

Appendix A. Course Structure



Appendix B. Useful Links and Resources

- The Hasso Plattner Institute: https://hpi.de/en/research/publications.html
- Student Project Challenge: https://www.weforum.org/
- User persona template: https://www.invisionapp.com/inside-design/user-persona-template/
- Usability testing: https://www.usability.gov/
- Journey map: https://www.nngroup.com/articles/customer-journey-mapping/
- User testing: https://www.nngroup.com/topic/user-testing/
- Story Board Guide by Rachel Krause: https://www.nngroup.com/articles/storyboards-visualize-ideas/
- UX cheat sheet: https://www.nngroup.com/articles/ux-research-cheat-sheet/
- Design tools: https://www.ideo.com/tools
- Swift Programming Language Guide: https://docs.swift.org/swift-book/LanguageGuide/TheBasics.html
- Swift iOS Features: https://developer.apple.com/swift/
- Xcode IDE Resources: https://developer.apple.com/xcode/resources/

Information Systems & Computing
Academic Professionals

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 2574-3872