Project Part A

● Graded

Group

OLIVIA ZHANG
YITENG JIANG
MIKE SHI
...and 2 more
View or edit group

Total Points

10 / 10 pts

Question 1

Motivation/Concepts

2 / 2 pts

✔ **– 0 pts** Correct

– 0.5 pts The motivation is good but the concept is unclear to me because the main feature heavily relies on the RL model training, which is a bit out of scope of our course project.

+ In general, ride-sharing sounds like a good idea to me and I think the workload is reasonable.
+ The technology stack and the capability of team members also match well.
* It would be better if the team members could clarify what third-party libraries can be used to facilitate the development. Reusing existing solutions for non-critical functionalities can help the team focus on the main features and deliver a more complete product faster.
* You will need to have an MVP (minimum viable product) by the end of part C for your presentation and demo. So you may want to start implementation early and plan accordingly.

Question 2

Epics

4 / 4 pts

✔ **– 0 pts** Correct

– 0.5 pts Epics are more related to developers but have relatively weak connection with users. For example, the AI training part may be not directly related to users.

Question 3

Feasibility and Contribution

2 / 2 pts

✔ **– 0 pts** Correct

Question 4

Technology Stack

2 / 2 pts

✔ **– 0 pts** Correct

No questions assigned to the following page.

Swift Link

Team MAX

Team members:

Yiqiao (May) Wang (905319079)

Olivia Zhang (705377225)

Rick Yang (405346443)

Yiteng Jiang (805577865)

Mike Shi (005592011)

GitHub repository: https://github.com/zhtyolivia/UCLA-CS130-Project

Question assigned to the following page: 1

Motivation/Concept

In California, especially in Los Angeles, there exists a pressing need for ridesharing as an efficient, cost-effective, and environmentally friendly transportation option. The problem is multifaceted. First, people who drive alone often seek people to share rides with in order to use the carpool lanes, which usually requires at least two passengers. Second, there are high costs associated with solo long-distance travel, such as from Los Angeles to San Francisco, and people can split costs by ridesharing. Thirdly, single-occupancy vehicles increase the emission of CO2, negatively impacting the environment. All of the above necessitates creating greater accessibility to finding the appropriate person to share rides with.

There are several existing solutions to the problem. Ride-hailing companies – most notably Uber Technologies, Inc. and Lyft, Inc. – provide shared rides by connecting riders who share similar routes and offering them a discounted price. However, these ride sharings services suffer from two main problems. Firstly, only riders can share a ride, and an ordinary driver (say someone driving to work) cannot share their car with another rider. Secondly, even with rider sharing, traveling for a long distance with these ride-hailing services is still expensive and in some cases impossible.

Our proposed application, **Swift Link**, aims to address these challenges. This website will serve as a dynamic platform, enabling users to connect with others for shared rides. The core premise is simple yet powerful: facilitate ride sharing for people traveling to or from Los Angeles from other parts of California, especially benefiting those who do not own a car, who need more riders to use the carpool lanes, and/or who seek people to split fuel expenses. To build such a platform, we will incorporate the following core features:

1. **User setting up and managing an account**. Rideshare-seekers will be able to create and manage their accounts, including uploading a user icon, resetting password, editing phone number, etc.
2. **User initiating a rideshare**. Users will be able to post about a rideshare they initiate, with relevant information such as starting and ending locations and preferred number of riders.
3. **Rideshare matching and searching.** Users will be able to search all rideshare-seeking posts by time or location. Upon searching, they should be able to view the details of the ride including time, location, and available spots.
4. **User registering to join a rideshare**. Users will be able to send a join request that includes information like number of passengers and a message to the driver to join a rideshare they like.
5. **User Accepting/declining a join request**. User (driver) will be able to accept or decline a pending join request from a passenger.

Questions assigned to the following page: 1, 2, and 4

6. **User exchanging information after matching**. Users will be able to exchange information with other users on the same share (driver and passenger) once a rideshare is successfully created.
7. **User quitting a rideshare**. Users can stop a rideshare they initiated or quit a rideshare they joined.
8. **User receiving notification upon status change**. All users/drivers should receive notification when there is a change of status to their ride (pending/processed join requests, canceled ride etc.)

To summarize, Swift Link is designed to bridge the gap in the current ride-sharing landscape by offering a user-friendly solution. It caters specifically to the needs of Californians, especially those in and around Los Angeles, who are looking for ridesharing. In the next seven weeks, our team will focus on implementing this idea by building an MVP of the product, thereby offering a solution to the ridesharing need in California.

Links to Epics

Description of Epic	**Link**
As a user, I would like to set up and manage my account.	https://github.com/zhtyolivia/UCLA-CS130-Project/issues/1
As a user, I would like to initiate a rideshare.	https://github.com/zhtyolivia/UCLA-CS130-Project/issues/2
As a user, I would like to register to join a rideshare.	https://github.com/zhtyolivia/UCLA-CS130-Project/issues/3
As a user, I would like to receive email notifications.	https://github.com/zhtyolivia/UCLA-CS130-Project/issues/4

Technology Stack

Frontend Development: We will be using React.js for our frontend. This library allows us to create components and will make our system very modular. This along with the highly dynamic and sophisticated User Interface we are building, made React.js a perfect match. React.js is also very good at creating stateful components, which is something crucial for our project as our users will be posting, and looking for ride shares through our system. We will also be using Redux to manage the state of our components, Axios to deal with HTTP requests, and a variety of other JavaScript libraries like Lodash to manipulate the data we will be using. Finally, we will use Jest to perform testing.

Backend Development: Our back-end is built around Node.js. It works in conjunction with Express.js which is a lightweight framework that helps quicken the development of robust

Question assigned to the following page: 3

server side applications. The flexibility of Express.js is needed in order to build out advanced user account management and ride coordination features that will be key. We'll also be using Passport.js to secure user authentication, and Mongoose for interaction with our MongoDB database.

Database Management: We will be using MongoDB as the database since we have experience with it and it provides a flexible import of data. MongoDB has a schema-less design that allows the Agile incorporation of change, unlike a traditional database such as SQL which has a structured architecture whereby a change to a required throws up a few other required changes. Additionally, MongoDB possesses capable querying functionality and is very fast, which is essential for a responsive ride service.

Deployment and CI/CD: For our deployment, we are thinking of Heroku or AWS. We are choosing them because they are well-known and trusted. Heroku has an easy to setup deployment and CI/CD, which will help us with this project. For our CI, we are contemplating using Github Actions. This tool will help us automate our testing and deployment workflow inside of our GitHub repository. This will be helpful throughout our development of the project.

Overall Technology Stack: In conclusion, the overall technology stack of React.js, Node.js, Express.js, and MongoDB work well together to make SwiftLink a complete and efficient platform. Along with the libraries and the ways of deployment we have picked, our project is in an ideal setup to meet the goals that we have set during this project. I am confident in what we have chosen for our website and I feel that it is a great setup for a great project. With our skills as a team, with our deadlines and work distributed, using this stack on our project we should have a mini version of a complete ride-sharing application in seven weeks that has high-quality code.

Feasibility

To enable our team to finish the project within the restricted time period, we analyzed the scope of the project taking our skill sets into account. As a team, we have three experienced backend developers and two experienced frontend developers. This would enable us to implement an MVP of our web application. Moreover, we will also adhere to the SCRUM process – we will hold standup meetings twice a week and have sprint retrospectives and reviews. In this way, we can make sure that everyone is on the same page, and we address issues and blockers in a timely manner.

We assigned two of our team members to focus on the frontend development using React. React's component-based architecture fits our idea for building a responsive user interface which is a ride-sharing platform. React provides efficient update and rendering technology that ensure a smooth user experience of the website. For backend development, we have three members working with Node.js and Express.js. Node.js and Express.js are useful tools to build server-side applications quickly and efficiently. For the database component of our project, we chose

Question assigned to the following page: 3

MongoDB because our team is more familiar and has more experience with this particular database platform. MongoDB is known for its flexibility and scalability by enabling us to change and expand the database structure as our project develops, which is an excellent fit for our application requirements.

To summarize, following a SCRUM process, reasonably allocating work among members, and using a suitable technology stack all ensure the feasibility of the project. By allocating sufficient time to complete each part of the project, we follow a reasonable timeline, further safeguarding the completion within the seven-week period.

Capability

Frontend Development: Frontend development of our application is an important part to ensure user friendliness and aesthetics. Both Ethan and Olivia will collaborate in the frontend development process. Ethan has previous experience using React and sending requests via Node.js during his work on Android application development. On Olivia's side, she has taken the CS97 course, in which she contributed to the frontend development of a web application using React. Also, both Ethan and Olivia are able to quickly learn new technologies that will be used in this project, given their experiences with multiple programming languages and frameworks. As for UI design, Mike has previous experience of UI design using Figma. Therefore, the combined skills from Ethan, Olivia, and Mike lay a solid foundation for the frontend development.

Backend Development: Rick, Mike, and May would collectively own the backend development. Rick has two intern experiences in Alibaba and Bytedance, which has given him a good sense of development in a team. As for web development, he has hands-on full-stack experience in building a web application in CS144, he built a backend server and implemented RESTful API to handle feed posts. More significantly, Rick has experience in mobile IOS development, and his familiarity with user interface and experience design would be helpful to the project, as well as responsive design for web applications. Mike will benefit the team through applying his technical program manager skills, as well as his experience with developing the backend in CS35L, which ensures the scalability and reliability of the application's backend structure. In addition, May has experience building an online trading platform for UCLA students showcases her initiative and experience in creating complex backend systems, with a working knowledge of many programming languages including C++, Java, and Python. With her practical experience and technical knowledge, May will be able to effectively contribute to the construction of the backend development within the project timeframe. It is worth mentioning that the fact that these three people all have backend development experience using Node.JS, the team would have relatively low learning costs, which is more conducive to rapid development. They would also be responsible for handling the automation for deploying the application.