**[Byte Builders]**

Val Estibeiro, Vushma Ahmad, Adepeju Awoyemi, Emaan Dar

IS 448: Markup and Scripting Language

**D1: Team Charter Outline**

## **Roles:**

Group leader: Valerie Estibeiro

Lead Analyst: Vushma Ahmad

Lead Programmer: Emaan Dar

Test Leads: Adepeju Awoyemi

**Team Purpose:**

* What does a team do?
  + The team is working together to create an application that addresses an issue within the UMBC (University of Maryland - Baltimore County) environment.
  + In order to successfully complete this task, our team will develop a web application that will exhibit complexity, integration of forms, and interaction with a database. There will be client-side functionality with JavaScript and Ajax. Additionally, there will be server-side interactions using PHP tailored to the University’s needs.
* For whom does the team do this activity/project?
  + Our team will work to address issues within UMBC in order to help students and faculty within UMBC.
* Why does the team do this activity/project?
  + Our team is a startup company that works on innovative concepts and we would like to address and resolve specific issues within the UMBC environment.

**Duration and Time Commitment:**

* Amount of time the team will work together?
  + Probably an hour and a half twice a week depending on the availability of each member
  + Maybe more added work time if done online
* Estimated amount of time that will be dedicated weekly?
  + Depending on availability and phases, 2-4 hours weekly
* When will the project end?
  + Deliverables are due throughout the semester but the final demo is due on the 14th of May

**Team Members:**

* Emaan Dar
* Valerie Estibeiro
* Vushma Ahmad
* Adepeju Awoyemi

**Team Practices:**

* How will the team work together? What strategies will be followed?
  + Each team member is expected to contribute ideas, feedback, and insights. Additionally, open communication is encouraged to ensure that everyone’s views and ideas are considered
* How often will you meet?
  + At a minimum, our team will meet twice a week to distribute tasks, work on the assignment target, and/or check in with each other to see how far along we all are and if anybody needs help for a specific part of the assignment.
* How will you divide coding and other activities?
  + The components of the coding/other assignments will be evenly distributed amongst the team members to ensure that all of the responsibility does not fall on one or two members.

**Team Performance Assessment:**

* List key areas of performance needed for team success along with means of measuring progress?
  + Key areas of performance for team success include communication, collaboration, and leadership. These areas of performance will be measured using surveys/peer evaluations and weekly check-ins in which members can comment on issues/concerns that they have regarding the project.
* How will the team measure that it has achieved its goals?
  + Prior to the submission deadline, the team will meet online to go over the requirements and goals of the project. In this meeting, the Project Lead will go over all of the requirements listed on the rubric. Next, each team member will respond stating whether their section is completed and the goals have been achieved.
* Identify a policy for a fair share of the workload?
  + The team will meet online to evenly distribute the workload. Additionally, the workload will be distributed as soon as the deliverable requirements are assigned. Furthermore, if a team member feels as though the workload is not fairly distributed, they have the opportunity to voice their concerns.
  + If a team member feels as though the workload was not evenly distributed after the initial meeting, it is their responsibility to reach out to the team leader. Next. the team leader will set up a meeting to discuss the concerns and redistribute the tasks to the rest of the team.

**Communication Strategies:**

* Process of sharing information both within the team and outside the team?
  + Our primary method of communication is through texting in a group chat in order to share communication information such as meeting time and place
  + Our secondary form of communication is through email in order to share information or code with each other

**Decision Making and Accountability:**

* Process for how decisions are made in team and communicated?
  + We simply get on an online call to go over things that need to be addressed as a group and make a compromise if needed
  + Texting updates about code in our group chat as well to
* How do team members hold each other responsible for commitments made to the team?
  + Verbal communication and or online communication
* How to deal with disparity in workload distribution?
  + Making sure the work is being evenly distributed in the first place, and notifying each other if one member feels like they are doing more work
* How the team will evolve in case problems arise such as team members' lack of following proper decorum and respect during team communications, inability by team member or team to meet deadlines in a timely manner?
  + We all are committed and communicative with each other so I am sure there is no area of issue when it comes to lack of respect or comms. But if there is an issue then talking in person usually helps
* How will workload shift in case of an indisposed team member?
  + If one of our team members is feeling unwell or has some sort of extenuating circumstance we will try to divvy up the work and make sure that they are able to do what they can while the rest can try and take on what they can

**Resources:**

* Tangible materials and organizational support the team needs to accomplish its goals?
  + In our group chat, the team leader will issue weekly reminders for upcoming due dates and responsibilities.
  + There will be a physical copy of the team charter in each team member’s backpack. This can serve as a reminder of the expectations we have as individuals and as a team.

**Signature Page:**

Each team member should sign the charter. Serves as a contract/agreement between team members?

* We are all committed to this project and will work together to solve any issues and make our demo follow all the requirements needed as well as meeting up at designated times to ensure we communicate properly

**Signature of Team Members:**

Valerie Estibeiro,

Vushma Ahmad,

Emaan Dar,

Adepeju Awoyemi

**Project Proposal**

* parking
* dorms
* smart schedule (Ade)-> automatically adjust the schedule so classes don't overlap
* Retriever ride share -> carpooling for UMBC and faculty
* Room finder (Val) -> what rooms are available -> classroom, study spaces/rooms, meeting places -> search for available rooms by availability and time can reserve live availability
* UMBC dining menu -> daily updated food menu/items, filter based on dietary restrictions (halal, vegan, vegetarian)
* Live shuttle tracking -> notification of delays, live tracking
* Umbc weather (Emaan) -> current weather and forecast for UMBC
* Study buddy finder (Vushma)-> People put their availability and subject so that they could find a study buddy

Proposal #1: Room Finder (Val)

Expanding the room finder idea:

1. Need for the App

* This application can help students or even faculty find rooms that are free to use and available
* With this application, students can find available rooms to study and do work in instead of interrupting other classes

2. Innovation & Creativity

* With this app, students can figure out the schedules of availability for any room around the whole campus with real-time class schedules

3. Technologies Used

* HTML & CSS → For structuring and styling the web app interface.
* JavaScript & Ajax → To fetch live room data from APIs and update the UI dynamically.
* PHP & MySQL → To store user preferences (e.g., availability alert, notification settings).

4. Use Cases (Each one involves user input & database interaction)

1. Searching for available rooms
   * User Input: The app automatically detects the current location (UMBC) or allows manual selection.
   * System Process: Fetches real-time weather data via API call and displays temperature, wind, and precipitation details.
   * Database Interaction: None (API fetch only).
2. Reserve a room
   * User Input: User selects a date range for a forecast.
   * System Process: Retrieves forecast data from the API and displays results.
   * Database Interaction: None (API fetch only).
3. Room availability Changes
   * User Input: Users can set alerts for weather conditions (e.g., "Notify me if the temperature drops below 30°F").
   * System Process: Stores the user’s preferences in a database.
   * Database Interaction: Compares real-time weather updates to stored alert conditions and triggers a notification.
4. Find a Study Group or Open study room
   * User Input: Users select a past date to view historical weather conditions.
   * System Process: Queries the database for past weather records and displays results.
   * Database Interaction: Retrieves stored historical data from MySQL.
5. Report a room issue
   * User Input: Users select notification preferences (email, text, or in-app).
   * System Process: If UMBC announces a class delay/cancellation due to weather, a notification is sent.
   * Database Interaction: Stores user preferences & checks for weather-related class updates.

Proposal #2: UMBC Weather (Emaan)

Expanding the UMBC Weather App Idea:

1. Need for the App

* UMBC students, faculty, and staff often rely on general weather apps that do not provide campus-specific weather alerts or customized notifications related to university events.
* This app would centralize real-time weather updates, severe weather alerts, and forecast details tailored for UMBC's location.
* It can integrate with UMBC systems to provide alerts about class cancellations or delays due to weather.

2. Innovation & Creativity

* Unlike generic weather apps, this app would provide:
  + Live weather conditions specifically for UMBC’s campus.
  + Real-time alerts for weather-related campus disruptions (e.g., “Heavy rain expected at 3 PM. Expect parking lot flooding.”).
  + Custom user settings to receive alerts for specific times or certain weather conditions.
  + Interactive maps with campus-specific temperature zones.

3. Technologies Used

* HTML & CSS → For structuring and styling the web app interface.
* JavaScript & Ajax → To fetch live weather data from APIs and update the UI dynamically.
* PHP & MySQL → To store user preferences (e.g., weather alerts, notification settings).
* API Integration → Uses weather APIs (e.g., OpenWeatherMap) to get real-time data.

4. Use Cases (Each one involves user input & database interaction)

1. Checking Current Weather
   * User Input: The app automatically detects the current location (UMBC) or allows manual selection.
   * System Process: Fetches real-time weather data via API call and displays temperature, wind, and precipitation details.
   * Database Interaction: None (API fetch only).
2. 5-Day Forecast Lookup
   * User Input: User selects a date range for a forecast.
   * System Process: Retrieves forecast data from the API and displays results.
   * Database Interaction: None (API fetch only).
3. Custom Weather Alerts
   * User Input: Users can set alerts for weather conditions (e.g., "Notify me if the temperature drops below 30°F").
   * System Process: Stores the user’s preferences in a database.
   * Database Interaction: Compares real-time weather updates to stored alert conditions and triggers a notification.
4. Historical Weather Data for UMBC
   * User Input: Users select a past date to view historical weather conditions.
   * System Process: Queries the database for past weather records and displays results.
   * Database Interaction: Retrieves stored historical data from MySQL.
5. Weather & Class Cancellation Notifications
   * User Input: Users select notification preferences (email, text, or in-app).
   * System Process: If UMBC announces a class delay/cancellation due to weather, a notification is sent.
   * Database Interaction: Stores user preferences & checks for weather-related class updates.

Proposal #3: Smart Schedule (Ade)

Smart Schedule Application

1. Need for the Application:

* **Scheduling Conflicts:** Students at UMBC often face issues with overlapping classes when registering.
* **Manual Adjustments:** Students currently have to check for conflicts and adjust their schedules manually.
* **Time-Consuming & Frustrating:** The process of resolving schedule conflicts takes time and adds stress.
* **Delays in Course Completion:** Overlapping classes can prevent students from enrolling in required courses on time.
* **Solution:** The "Smart Schedule" application will automatically adjust schedules to prevent conflicts.
* **Benefits:** This tool will make scheduling easier, more efficient, and reduce student stress, improving their academic experience.

1. Innovation and Creativity:

* **Conflict Detection:** The app analyzes a student’s selected courses to identify scheduling conflicts.
* **Optimized Scheduling:** It suggests the best possible schedule based on available class sections and timings.
* **Personalization:** The app considers student preferences, such as morning or evening classes, to tailor schedules.
* **Smart Automation:** It eliminates the need for manual adjustments, making scheduling more efficient.
* **Enhanced Course Management:** Provides a smarter way for students to organize their academic schedules seamlessly.

1. Technologies Used:

The Smart Schedule application will use the following technologies:

* **HTML & CSS:** To design a user-friendly interface where students can input their course preferences and view their schedules.
* **JavaScript & Ajax:** To enhance client-side interactions, making the application responsive and interactive.
* **PHP:** To handle server-side logic and process student course requests.
* **MySQL:** To store and retrieve course information, student preferences, and scheduling data.
* **Ajax:** To enable real-time schedule adjustments without requiring page reloads.

1. Use Cases:

* **Input Course Preferences**
  + **User Input:** The student selects their desired courses and inputs personal preferences (e.g., preferred time slots, days off, etc.).
  + **System Process:** The application retrieves available class sections from the database and checks for scheduling conflicts.
* **Automatic Conflict Detection and Adjustment**
  + **User Input:** The student submits their initial schedule.
  + **System Process:** The application scans for time conflicts and suggests alternative sections or arrangements to resolve overlaps.
* **Real-Time Schedule Updates**
  + **User Input:** The student updates their preferences or adds a new course.
  + **System Process:** Using Ajax, the application immediately checks for conflicts and updates the schedule without requiring a page refresh.
* **Final Schedule Confirmation**
  + **User Input:** The student reviews and confirms the suggested schedule.
  + **System Process:** The application saves the finalized schedule in the database and provides a downloadable/printable version.

Proposal #4: Study Buddy Finder (Vushma)

1. Need for the App
   * The purpose of this application is to allow UMBC students to use an app in order to find a study buddy
   * The need for this application is that it saves time for students by eliminating the hassle of manually finding a study buddy through classroom interaction
   * Another need for this application is that it can improve the academic performance of students by helping them find study partners based on specific subjects and schedules
2. Innovation & Creativity
   * The UMBC Studdy Buddy Finder App would:
     + Have a section in the application to allow students to put the subject that they are struggling with and match them up with another student who is excelling in the subject.
     + Allow each student/user to input their availability to avoid scheduling conflicts
     + Be Not only able to find a study buddy, but also be able to find other students struggling or excelling in the subject in order to create a study group
     + Book study rooms through the app with a study buddy/group
3. Technologies Used
   * HTML & CSS → This will be used to structure the web page (HTML) and style/design the webpage (CSS)
   * JavaScript → Find study buddies based on subject and schedule
   * PHP & MySQL → Keep track of students/group profiles (MySQL) & match the students to study buddies based on their subject and schedule (PHP)
   * Ajax → Display study session schedules without having to refresh the page
4. Use Cases
   * Profile Creation
     + User Input → Information such as their name, email & password, subject, availability, and if they need help or want to help other students
     + System Process → Create and save the user profile into the database
   * Finding a Study Buddy
     + User Input → A specific subject, availability, and if they need help or want to help other students
     + System Process → Query the database to find matching users
   * Scheduling a Study Session
     + User Input → date and time for session
     + System Process → Check for scheduling conflicts in the database and update the students' calendars
   * Study Resource Section
     + User Input → Users can upload study materials such as PDFs, slides, and notes
     + System Process → The system will store the files in the database and allow for real-time updates