



Republic of the Philippines  
Western Mindanao State University  
**COLLEGE OF COMPUTING STUDIES**  
Department of Information Technology  
Zamboanga City



## **ShopNow Mini Store**

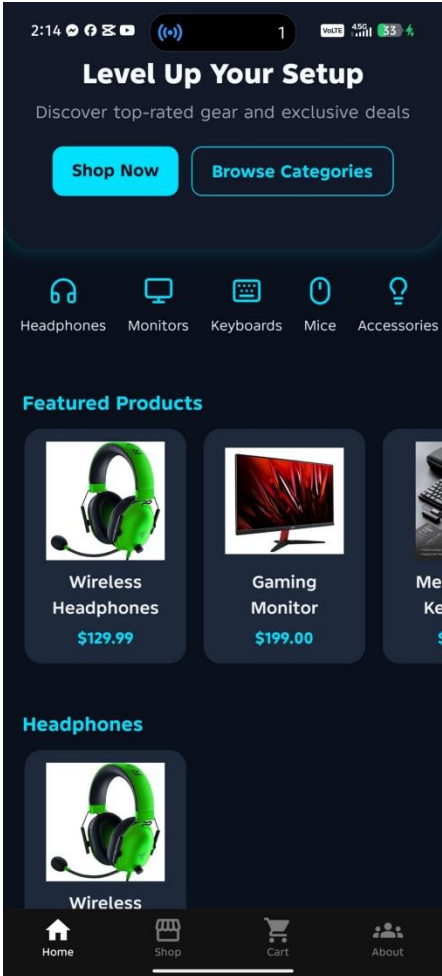
In Partial Fulfillment  
of the Requirements for the Degree  
Bachelor of Science in Information Technology

By:  
Ignacio, Rod Angelo Benedict V.  
Madrado, Alexandra Steffi Marie M.  
Remillate, John Zander S.  
Sarmiento, Ira C.

Submitted to:  
Mr. Rey Saavedra

Date of Submission:  
November 10, 2025

1. Home Screen/Tab



Code:

```
export default function HomeScreen() {
  const router = useRouter();
  const scrollRef = useRef(null);
  const categorySectionRef = useRef(null);
  const [categoryY, setCategoryY] = React.useState(0);

  const handlePress = (id) => router.push(`/product/${id}`);

  const handleShopNow = () => {
    router.push("/(tabs)");
  };

  const handleBrowseCategories = () => void
  const handleBrowseCategories = () => {
    if (scrollRef.current) {
      scrollRef.current.scrollTo({ y: categoryY, animated: true });
    }
  };
};
```

Explanation:

- The function handleShopNow handles the Shop Now function, where if the user clicks the Shop Now Button, it will redirect them to the Shop Tab.
- The function handleBrowseCategories handles the Browse Categories, where if the user clicks the Browse Categories button, it will scroll down to the categories.

Code:

```
<View style={styles.section}>
  <ThemedText type="title" style={styles.sectionTitle}>
    Featured Products
  </ThemedText>
  <FlatList
    data={PRODUCTS.slice(0, 3)}
    horizontal
    showsHorizontalScrollIndicator={false}
    keyExtractor={(item) => item.id}
    contentContainerStyle={{ paddingHorizontal: 12 }}
    renderItem={({ item }) => (
      <TouchableOpacity
        style={styles.productCard}
        onPress={() => handlePress(item.id)}
      >
        <Image
          source={item.image}
          style={styles.productImage}
          contentFit="contain"
        />
        <ThemedText style={styles.productName}>{item.name}</ThemedText>
        <ThemedText style={styles.productPrice}>{item.price}</ThemedText>
      </TouchableOpacity>
    )}
  />
</View>
```

Explanation:

- This part of the Code handles the display of the featured products, where it only displays 3 products by using the `data={PRODUCTS.slice(0, 3)}`.

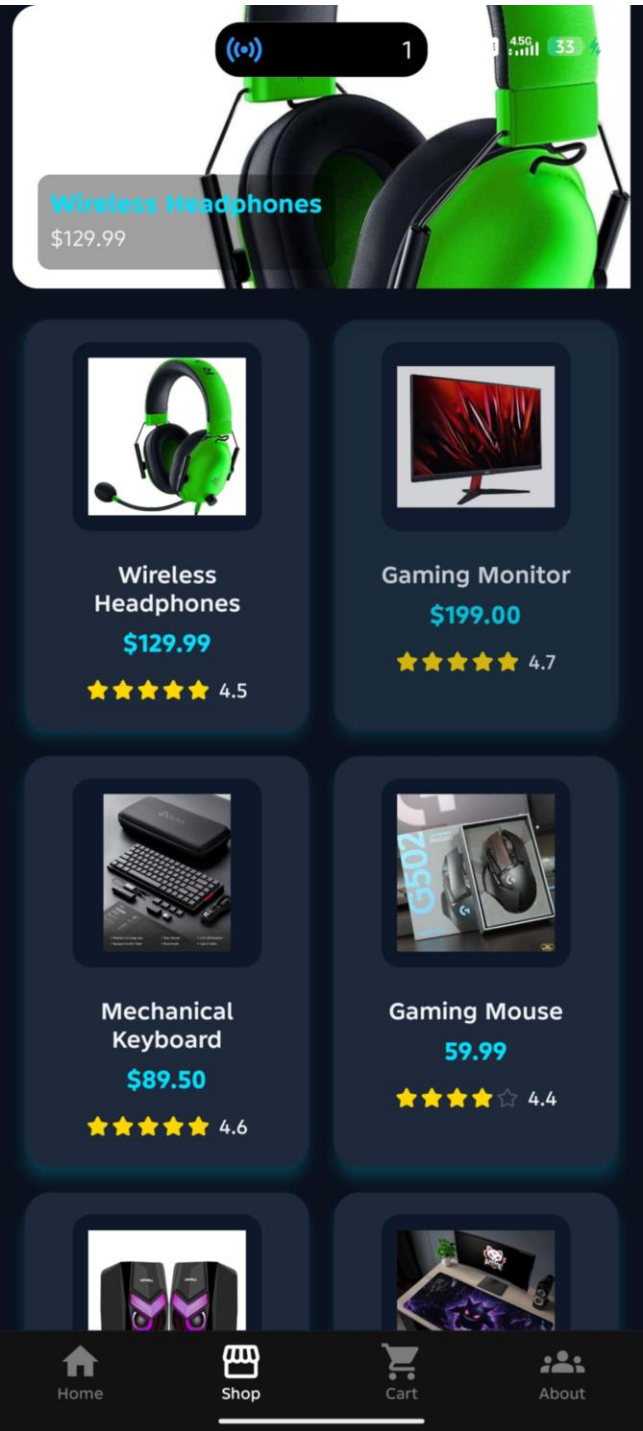
Code:

```
{categorizedProducts.map(
  (category) =>
    category.data.length > 0 && (
      <View key={category.title} style={styles.section}>
        <ThemedText type="title" style={styles.sectionTitle}>
          {category.title}
        </ThemedText>
        <FlatList
          data={category.data}
          horizontal
          showsHorizontalScrollIndicator={false}
          keyExtractor={(item) => item.id}
          contentContainerStyle={{ paddingHorizontal: 12 }}
          renderItem={({ item }) => (
            <TouchableOpacity
              style={styles.productCard}
              onPress={() => handlePress(item.id)}
            >
              <Image
                source={item.image}
                style={styles.productImage}
                contentFit="contain"
              />
              <ThemedText style={styles.productName}>
                {item.name}
              </ThemedText>
              <ThemedText style={styles.productPrice}>
                {item.price}
              </ThemedText>
            </TouchableOpacity>
          )}
        </FlatList>
      </View>
    )
  )}
```

Explanation:

- This part of the code handles the display of all products by categories by using the `.map` it will iterate over each element in the array, then it will display it.

2. Shop Screen/Tab



Code:

```
export default function ShopScreen() {
  const router = useRouter();
  const scrolly = useRef(new Animated.Value(0)).current;
  const promoRef = useRef(null);
  const [promoIndex, setPromoIndex] = useState(0);

  const promoItems = PRODUCTS.slice(0, 5);

  const handlePress = (id) => {
    router.push(`/product/${id}`);
  };

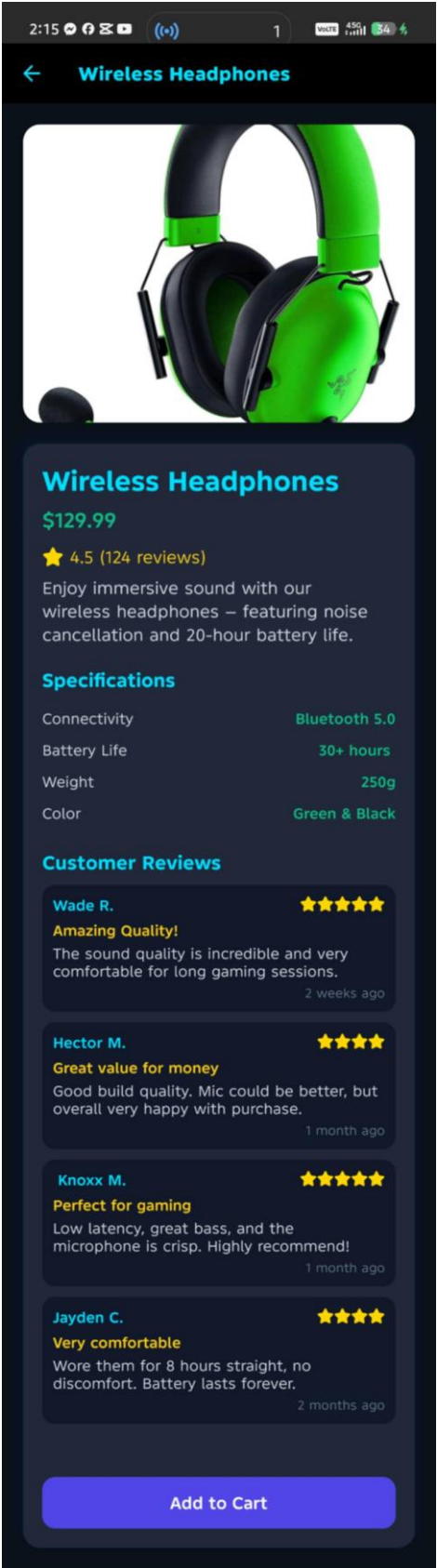
  useEffect(() => {
    const interval = setInterval(() => {
      const nextIndex = (promoIndex + 1) % promoItems.length;
      setPromoIndex(nextIndex);
      promoRef.current?.scrollToOffset({
        offset: nextIndex * width,
        animated: true,
      });
    }, 3000);

    return () => clearInterval(interval);
  }, [promoIndex]);
```

Explanation:

- The handlePress function handles the function for each product pressed will redirect the user to the Product Detail, where the stack comes in. If the user clicks any of the products, the product detail will stack, then when the back button is clicked, it will unstack the product detail.
- The use of useEffect here is to make the carousel in the shop screen, where it displays certain products, then it will scroll one by one automatically.

3. Product Detail Screen/Stack



Code:

```
export default function ProductDetail() {
  const { id } = useLocalSearchParams();
  const { cart, addToCart } = useCart();
  const router = useRouter();
  const [visible, setVisible] = useState(false);
  const fadeAnim = React.useRef(new Animated.Value(0)).current;
  const [popupMessage, setPopupMessage] = useState("");
  const product = PRODUCTS.find((item) => item.id === id);
  (parameter) message: any

  const showPopup = (message) => {
    setPopupMessage(message);
    setVisible(true);
    Animated.timing(fadeAnim, {
      toValue: 1,
      duration: 200,
      useNativeDriver: true,
    }).start();

    setTimeout(() => {
      Animated.timing(fadeAnim, {
        toValue: 0,
        duration: 300,
        useNativeDriver: true,
      }).start(() => setVisible(false));
    }, 1500);
  };

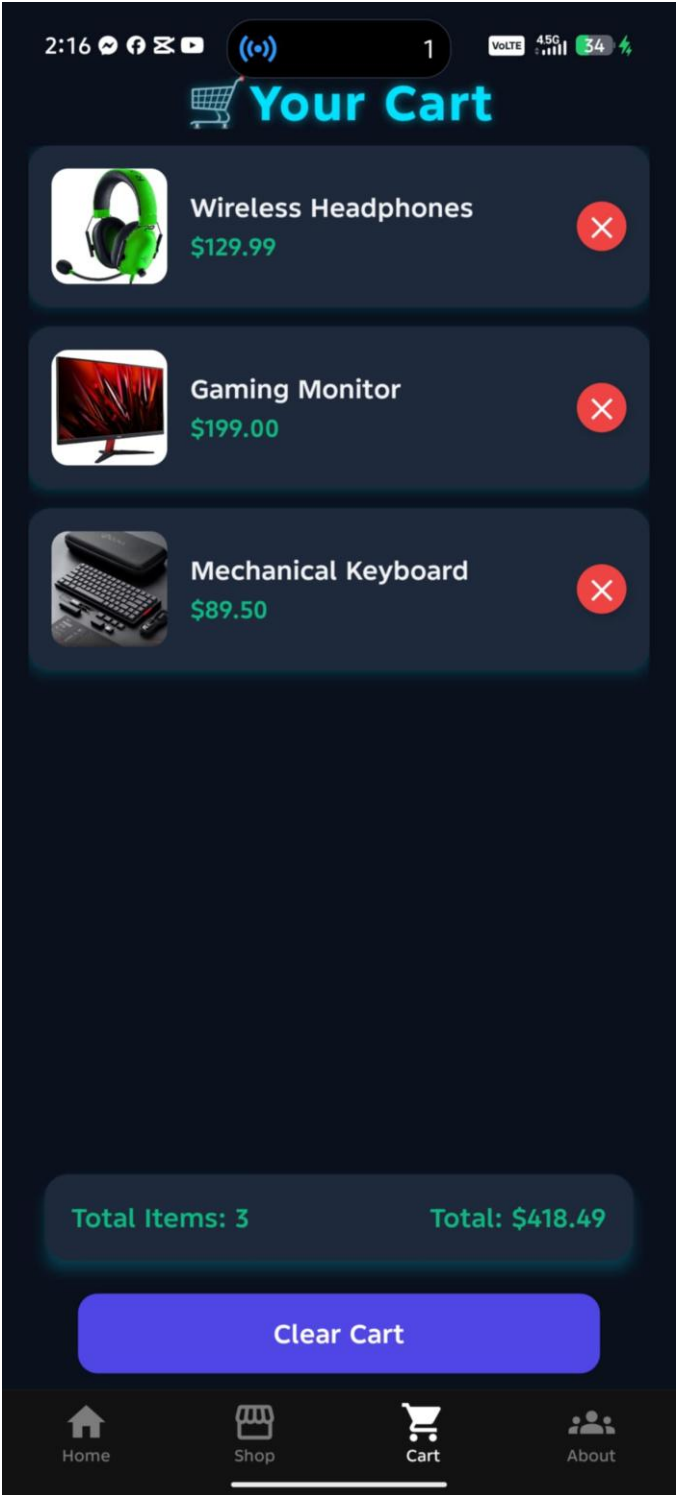
  const handleAddToCart = () => {
    const isAlreadyInCart = cart.some((cartItem) => cartItem.id === product.id);

    if (isAlreadyInCart) {
      showPopup("⚠ Item already in cart");
    } else {
      addToCart(product);
      showPopup("🛒 Added to cart!");
    }
  }
}
```

Explanation:

- The showPopup handles the appearance and animation of the pop-up when clicking the add to cart button
- The handleAddToCart handles the checking of duplication of products, so it will avoid adding multiple copies of the same product to the cart. If there is already a product that is the same product in the cart, it will send a pop-up warning, but if not, then it will proceed with adding it to the cart.

4. Cart Screen/Tab



Code:

```
export default function CartScreen() {
  const { cart, removeFromCart, clearCart } = useCart();

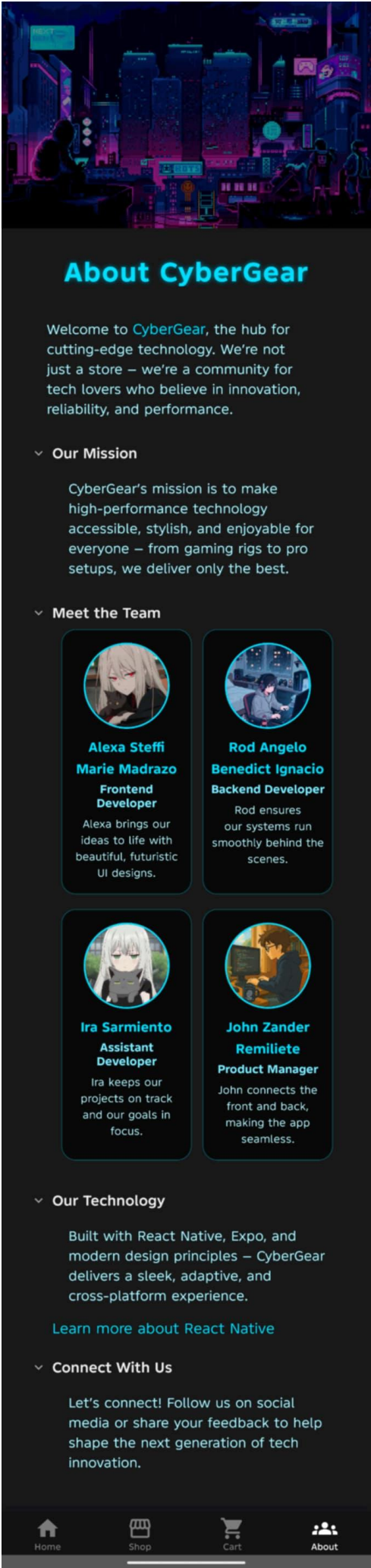
  const totalItems = cart.length;
  const totalCost = cart.reduce((sum, item) => {
    const price = Number(item.price.toString().replace(/^[^0-9.-]+/g, ""));
    return sum + price;
  }, 0);

  const renderItem = ({ item }) => (
    <View style={styles.card}>
      <Image source={item.image} style={styles.image} />
      <View style={styles.info}>
        <Text style={styles.name}>{item.name}</Text>
        <Text style={styles.price}>{item.price}</Text>
      </View>
      <TouchableOpacity
        style={styles.removeButton}
        onPress={() => removeFromCart(item.id)}
      >
        <Text style={styles.removeText}>X</Text>
      </TouchableOpacity>
    </View>
  );
};
```

Explanation:

- The totalItems handles the length of the items inside the cart.
- The totalCost computes the overall price of all the products inside the Cart.
- The renderItem handles the displaying of the products and adding additional buttons like the X button for removing a specific product and the Clear Cart button for clearing all the products inside the Cart.





Code:

```
const teamMembers = [
  {
    role: 'Frontend Developer',
    image: require('@assets/images/team/team2.jpg'),
    description: 'Alexa brings our ideas to life with beautiful, futuristic UI designs.',
  },
  {
    name: 'Rod Angelo Benedict Ignacio',
    role: 'Backend Developer',
    image: require('@assets/images/team/team1.jpg'),
    description: 'Rod ensures our systems run smoothly behind the scenes.',
  },
  {
    name: 'Ira Sarmiento',
    role: 'Assistant Developer',
    image: require('@assets/images/team/team4.jpg'),
    description: 'Ira keeps our projects on track and our goals in focus.',
  },
  {
    name: 'John Zander Remiliete',
    role: 'Product Manager',
    image: require('@assets/images/team/team3.jpg'),
    description: 'John connects the front and back, making the app seamless.',
  },
];

const fadeAnim = useRef(new Animated.Value(0)).current;

useEffect(() => {
  Animated.timing(fadeAnim, {
    toValue: 1,
    duration: 1200,
    useNativeDriver: true,
  }).start();
}, []);
```

Explanation:

- teamMembers Stores the details of each team member.
- use of useEffect here for animation of the text “About CyberGear”

Code:

```
<ThemedText
  type="title"
  style={{
    fontFamily: Fonts.rounded,
    color: '■ #00E0FF',
    textShadowColor: '■ #00E0FF90',
    textShadowOffset: { width: 0, height: 0 },
    textShadowRadius: 15,
    fontSize: 30,
    letterSpacing: 1,
    textAlign: 'center',
  }}
>
  About CyberGear
</ThemedText>
</Animated.View>

<ThemedText style={styles.storyText}>
  Welcome to <ThemedText style={{ color: '■ #00E0FF' }}>CyberGear</ThemedText>, the
  hub for cutting-edge technology. We're not just a store – we're a community
  for tech lovers who believe in innovation, reliability, and performance.
</ThemedText>

<Collapsible title="Our Mission" containerStyle={styles.collapsibleContainer}>
  <ThemedText style={styles.storyText}>
    CyberGear's mission is to make high-performance technology accessible,
    stylish, and enjoyable for everyone – from gaming rigs to pro setups,
    we deliver only the best.
  </ThemedText>
</Collapsible>
```

Explanation:

- Display the about of the shop.
- having a collapsible component so the text can be hidden or shown.

Code:

```
<Collapsible title="Meet the Team" containerStyle={styles.collapsibleContainer}>
  <ThemedView style={styles.teamContainer}>
    {teamMembers.map((member, index) => (
      <Animated.View
        key={index}
        style={[
          styles.teamCard,
          {
            transform: [
              {
                scale: fadeAnim.interpolate({
                  inputRange: [0, 1],
                  outputRange: [0.9, 1],
                }),
              },
            ],
          },
        ]},
      <Image source={member.image} style={styles.teamImage} />
      <ThemedText type="defaultSemiBold" style={styles.teamName}>
        {member.name}
      </ThemedText>
      <ThemedText type="subtitle" style={styles.teamRole}>
        {member.role}
      </ThemedText>
      <ThemedText style={styles.teamDesc}>{member.description}</ThemedText>
    </Animated.View>
    )
    )}
  </ThemedView>
</Collapsible>
```

Explanation:

- Display all the members' details, including name, role, and description, using .map to iterate and display everything inside the array.